QUEUE USING LINKED LIST

Queue is a First In First Out [FIFO] data structure. In chapter 4, we have discussed about stacks and its different operations. And we have also discussed the implementation of stack using array, ie; static memory allocation. Implementation issues of the stack (Last In First Out - LIFO) using linked list is illustrated in the following figures.

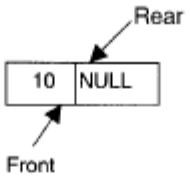

Fig. 5.16. push (10)

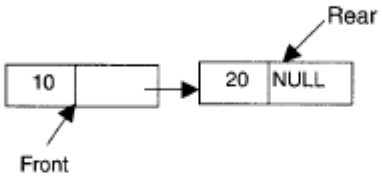

Fig. 5.17. push (20)

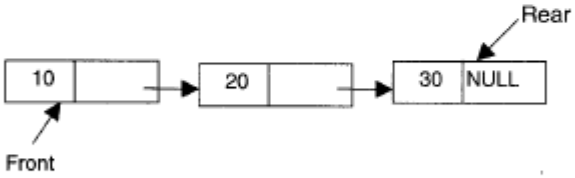

Fig. 5.18. push (30)

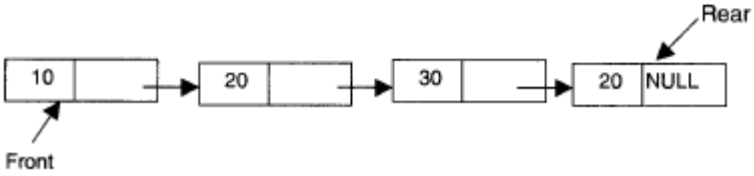

Fig. 5.19. push (40)

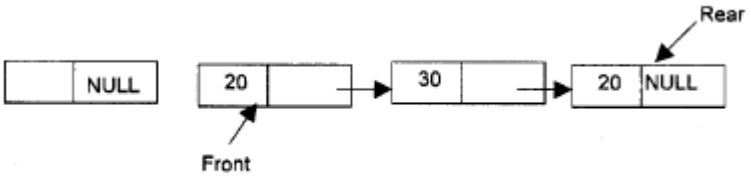

Fig. 5.20. X = pop() (*i.e.*; X = 10)

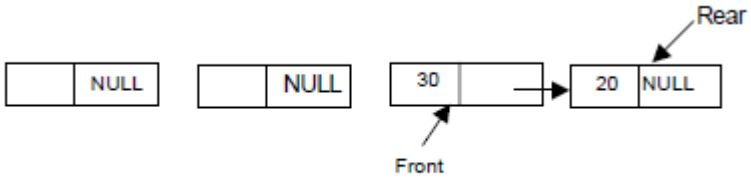

Fig. 5.21. X = pop() (*i.e.*; X = 20)

ALGORITHM FOR PUSHING AN ELEMENT TO A QUEUE

REAR is a pointer in queue where the new elements are added. FRONT is a pointer, which is pointing to the queue where the elements are popped. DATA is an element to be pushed.

1. Input the DATA element to be pushed
2. Create a New Node
3. NewNode → DATA = DATA
4. NewNode → Next = NULL
5. If(front is equal to NULL and rear is equal to NULL)
   (*a*) front = rear = NewNode
   (b) exit
6. rear → next = NewNode
7. rear = NewNode
7. Exit

ALGORITHM FOR POPPING AN ELEMENT FROM A QUEUE

REAR is a pointer in queue where the new elements are added. FRONT is a pointer, which is pointing to the queue where the elements are popped. DATA is an element popped from the queue.

1. declare temp = FRONT
2. If (FRONT is equal to NULL)
   (a) Display "The Queue is empty"
3. Else if (FRONT is equal to REAR)
   (a) FRONT = REAR = NULL
4. Else

(a) FRONT = FRONT → next

5. delete temp

6. Exit